Modelos de Diferencias en el Tiempo

Alvaro J. Riascos Villegas
Universidad de los Andes y Quantil

Nociembre de 2024

Contenido

Introducción

- Temporal Difference (TD) es una combinación de MC y PD.
- Como los métodos de Montecarlo usa simulaciones de episodios pero tambien hace bootstrap: la actualización de la función valor depende de valores estimados.
- Comenzamos con el problema de predicción de la función valor del estado (algoritmo TD(0)).
- Después continuamos con el problema de control (algoritmo SARSA).

Contenido

Bandido Multiarmado

- Los métodos de Montecarlo se pueden implementar de forma incremental.
- En el caso del Bandido Multiarmado:

$$Q_{t+1}(a) = Q_t(a) + \frac{I_{[A_t=a]}}{\sum_{i=1}^{t} I_{[A_i=a]}}(R_t(a) - Q_t(a)) \qquad (1)$$

- En el caso con decaimiento exponencial se utiliza un parámetro $\alpha \in (0, 1)$ y se utiliza la actualización:

$$Q_{t+1}(a) = Q_t(a) + \alpha(R_t(a) - Q_t(a)) \qquad (2)$$

- La forma general de esta estrategia es:

Nueva estimación $\leftarrow$ Estimación anterior + Tamaño del salto $\times$ (Recompensa − Estimación anterior)

- En el caso general de RL es fácil mostrar que:

$$V^{n+1}(s) = V^n(s) + \frac{1}{n+1}(G_{n+1} - V^n(s)) \tag{3}$$

- Esto sugiere este tipo de actualización incremental en el problema general de MDP.

$$V(s_t) \leftarrow V(s_t) + \alpha(G_t - V(s_t)) \tag{4}$$

y para la función valor del estado y acción:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(G_t - V(s_t, a_t)) \tag{5}$$

Contenido

Predicción Política usando Montecarlo: Implementación Incremental)

- Repasemos el algoritmo de primera visita para predecir la función valor dada una política usando Montecarlo.
- Obsérvece que se simulan episodios completos y después se actualiza la función valor.

Predicción Política usando Montecarlo: Implementación Incremental)

First-visit MC prediction, for estimating $V \approx v_\pi$

Input: a policy π to be evaluated

Initialize:

$V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$

$Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

Loop forever (for each episode):

Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless S_t appears in $S_0, S_1, \ldots, S_{t-1}$:

Append G to $Returns(S_t)$

$V(S_t) \leftarrow$ average$(Returns(S_t))$

- La versión incremental es cambiar la última linea por:

$$V(s_t) \leftarrow V(s_t) + \alpha(G_t - V(s_t)) \tag{6}$$

Evaluación de Política: TD(0)

- Hasta este punto hemos reptido Montecarlo pero con la actualización incremental.
- TD(0) capitaliza en la idea de bootstrapping de programación dinámica. La idea es, en la versión incremental, estimar G_t con $R_{t+1} + \gamma V(s_{t+1})$.
- La intución es la siguiente:

$$V_\pi(s_t) = E_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} ... \mid s_t] \quad (7)$$

$$= E_\pi[R_{t+1} + \gamma V_\pi(s_{t+1}) \mid s_t] \quad (8)$$

Evaluación de Política: TD(0)

- Luego TD(0) es el mismo algoritmo anterior sin simular episodios completos y en donde en la última línea se actualiza con:

$$V(s_t) \leftarrow V(s_t) + \alpha(R_{t+1} + \gamma V(s_{t+1}) - V(s_t)) \qquad (9)$$

- Obsérvese que al hacer esta estimación ya no es necesario simular episodios completos, solo un paso adelante.
- Por eso se dice que Temporal Difference hace bootstrapping y está inspirado en el método de Montecarlo.

Evaluación de Política: TD(0)

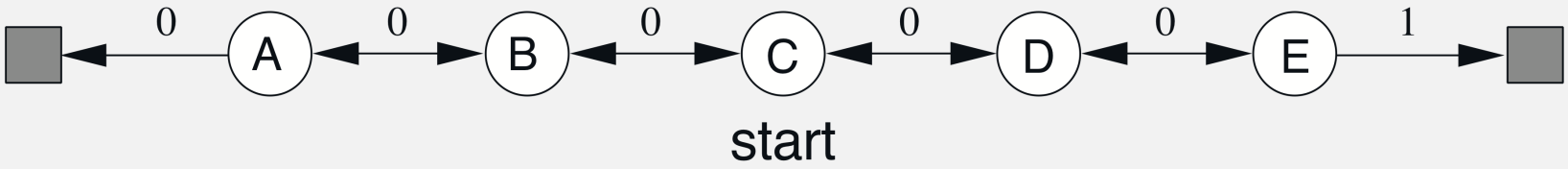
Tabular TD(0) for estimating v_π

Input: the policy π to be evaluated
Algorithm parameter: step size $\alpha \in (0, 1]$
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(\mathit{terminal}) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    $A \leftarrow$ action given by π for S
    Take action A, observe R, S'
    $V(S) \leftarrow V(S) + \alpha\big[R + \gamma V(S') - V(S)\big]$
    $S \leftarrow S'$
  until S is terminal

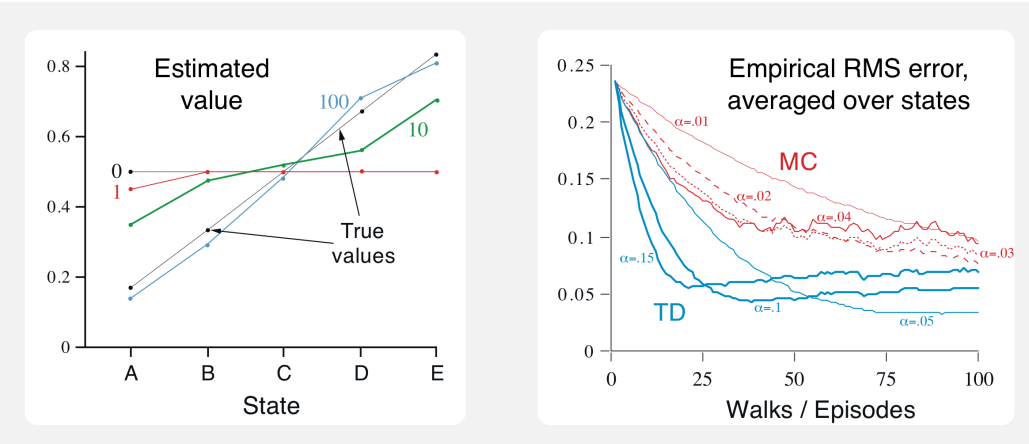
Ejemplo TD(0)

- Este es un proceso Markoviano de recompensas. Probabilidad de moverse en cualquier direccion es 0,5. Las recompensas como aparecen el grafo.



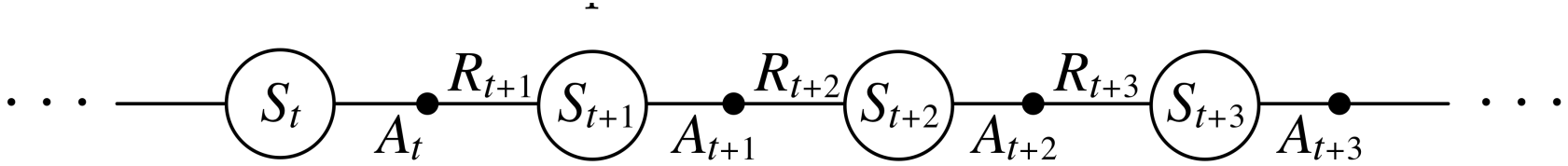
Ejemplo TD(0)

- La evaluacion verdadera de la funcón valor es la línea negra.
- Las demas líneas muestran la actualizacion de la función valor despúes de varios episodios. Con 100 episodios prácticamente descubre los verdaderos valores.



Contenido

SARSA



SARSA

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(\mathit{terminal}, \cdot) = 0$

Loop for each episode:
  Initialize S
  Choose A from S using policy derived from Q (e.g., ε-greedy)
  Loop for each step of episode:
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma Q(S', A') - Q(S, A)\big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

Ejemplo SARSA

- El objetivo es llegar de S a G.
- Existe un viento vertical que mueve el dron en cada columna como se identifica en la parte de abajo de la grilla.
- El eje Y muestra el número de pasos acumulados. Entre más episodios menos pasos para llegar al objetivo.
- Después de 8000 pasos se obtiene política que tarda 17 pasos en promedio para llegar. El óptimo en azul es 15.